Get your LSF batch jobs to start faster

Specifying a RUNLIMIT

You can minimize the time it takes for a general queue job to start running by defining a wall-clock time limit. Instead of explicity selecting a general queue (short, medium, long, xlong, xxl), just provide the RUNLIMIT argument to the bsub command. The syntax is '-W [hour:]minute'. This time limit should be based on a worst-case scenario since LSF will terminate the job if it exceeds the RUNLIMIT value. This is a real-world time measurement since we are not using any kind of normalization. The automatic queue selection feature will place your job in the appropriate general queue, eliminating any guesswork. Some examples:

```
yemi@rhel6-64g $ bsub -W 10 myVeryFastJob
Job <97451> is submitted to default queue <short>.

yemi@rhel6-64g $ bsub -W 60 myFastJob
Job <98011> is submitted to default queue <medium>.

yemi@rhel6-64g $ bsub -W 300 myLongerJob
Job <98365> is submitted to default queue <long>.
```

The shorter general queues have higher priority and automatic queue selection will put your job in the shortest compatible queue. If you use a dedicated (non-general) queue in your production environment, continue to specify the queue in your bsub command but add the '-W' option.

By supplying a RUNLIMIT, your jobs can start faster because they stand a better chance of using a feature called "backfill". An increasing number of users are now running large parallel jobs across multiple cores/slots. These parallel jobs can take a considerable amount of time to reserve all the cores they need to start. The scheduler will attempt to backfill (run smaller jobs on reserved cores) as long as the estimated start time of the bigger parallel job is not affected.

Providing a RUNLIMIT let's the scheduler know what the required time window for your job is. Without an explicit RUNLIMIT, the scheduler can only assume your job will run as long as the default RUNLIMIT for the queue - this default is often far greater than many jobs need! For example, the xlong queue currently has a RUNLIMIT default of 72 hours but queue statistics show the runtime average for jobs in this queue is currently ~2 hours.

Specifying a runtime estimate

The RUNLIMIT parameter may not provide enough flexibility for certain types of event processing. For example, the majority of jobs in a pipeline should complete on time, but some jobs get hung up and could take longer. We want to avoid having to resubmit longer running jobs that get killed off by the LSF scheduler. One solution is to provide a runtime estimate in addition to the RUNLIMIT. The argument syntax for the runtime estimate is '-We [hour:]minute'. The assumption is that most of the pipeline jobs will complete within the runtime estimate. The estimate value will also influence backfill. The scheduler will only terminate a job when it exceeds the RUNLIMIT, but not when it exceeds the estimate. Example of a job with a runtime estimate of 15 minutes and a RUNLIMIT of 1 hour:

```
yemi@rhel6-64g $ bsub -We 15 -W 1:00 processEventJob
Job <209332> is submitted to default queue <medium>.
```